Langara College
CPSC 2150 Lab
Nothing to submit: Lab will not be marked

Submit the file BST_fcts/**BST.cpp**

Given is the definition of a node in a Binary Search Tree BST

```
struct Node {
   Node* left;
   int val;
   Node* right;
};
```

Given is a BST with n unique (no duplicate) integer values such that every value of the left subtree is less than the node's value and every value of the right subtree is greater than the node's value.

part A)
Write a function that takes two such binary search trees p and q

```
bool sameVals(const Node* p, const Node* q);
```

and returns true if the binary search trees p and q have the same values (though not necessarily the same structure in the sense of the same children at the same level) and returns false otherwise.

Example
p points to the BST below, q points to the BST below, and sameVals(p, q) returns true

You **may** use temporary auxiliary memory but you must deallocate it yourself (to prevent memory leaks). Do not use global variables nor static variables. Do not use the STL.

part B)
If the BST p has n values and the BST q has m values, what is the complexity of your function sameVals?
Express your answer in big O notation and explain **why**.

Submit the file BST_fcts/**BST.cpp**

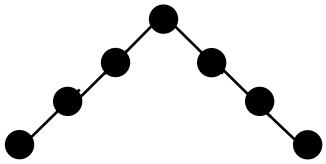
part A)
Write a function

```
bool heightBalanced(const Node* tree);
```

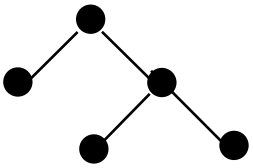
that determines if the binary search tree `tree` is height balanced. "A binary tree is *height balanced* if the difference in height of both subtrees of any node in the tree is either zero or one." (Drozdek, page 250)

Do **not** use global variables, linked lists, arrays, strings, static variables nor the STL for any part of the question.

Examples



not balanced



balanced

part B)
If the BST has n values, what is the complexity of your function heightBalanced? Express your answer in big O notation and explain **why**.

Submit under Midterm #2 the file BST/**BST.cpp**

Given is the definition of a node in a **binary tree**

```
struct Node {
   Node* left;
   int val;
   Node* right;
};
```

Do **not** use global variables, arrays, strings, static variables nor the STL for any question.

part A) (6 marks)
Write a function with the function prototype

```
int numNodes(const Node* tree);
```

that returns the number of nodes in the binary tree `tree`.

part B) (14 marks)
Given is a Binary Search Tree BST with n unique (no duplicate) integer values such that every value of the left subtree is less than the node's value and every value of the right subtree is greater than the node's value.

Write a function with the function prototype

```
int median(const Node* tree);
```

that returns the median of all the values in the BST `tree`.
The median is the middle value of a sequence of sorted values.
For `median`, assume that $n \geq 1$, i.e., there is at least one node in the tree.
If n is even, return either one of the middle values.
For example,
for a BST that stores the values 1 3 4 8 10 the median is 4
for a BST that stores the values 2 3 4 6 8 9 the median is either 4 or 6